Anchor View Allocation for Collaborative Free Viewpoint Video Streaming

Dongni Ren[*] *S.-H. Gary Chan*[*] *Gene Cheung*[†] *Vicky Zhao*[‡] *Pascal Frossard*[⋆]

[*]The Hong Kong University of Science and Technology, Hong Kong, China

[†]National Institute of Informatics, Tokyo, Japan

[‡]University of Alberta, Edmonton, AB, Canada

[⋆]École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

Email: {tonyren, gchan} @cse.ust.hk, cheung@nii.ac.jp, vzhao@ece.ualberta.ca, pascal.frossard@epfl.ch

Abstract

In free viewpoint video, a viewer can choose at will any camera angle or the so-called "virtual view" to observe a dynamic 3D scene, enhancing his/her depth perception. The virtual view is synthesized using texture and depth videos of two *anchor camera views* via depth-image-based rendering (DIBR). We consider, for the first time, collaborative live streaming of a free viewpoint video, where a group of users may interactively pull and cooperatively share streams of different anchor views. There is a cost to access the anchor views from the live source, a cost to "reconfigure" the peer network due to a change in selected anchors during view switching, and a distortion cost due to the distance of the virtual views to the received anchor views at users. We optimize the anchor views allocated to users so as to minimize the overall streaming cost given by the access cost, reconfiguration cost and view distortion cost.

We first show that, if the reconfiguration cost due to view switching is negligible, the view allocation problem can be optimally and efficiently solved in polynomial time using *dynamic programming*. For the case of non-negligible reconfiguration cost, the problem becomes NP-hard. We thus present a locally optimal and *centralized algorithm* inspired by Lloyd's algorithm used in non-uniform scalar quantization. We further propose a *distributed algorithm* with convergence guarantee, where each peer group independently makes merge-and-split decisions with a well-defined fairness criteria. Simulation results show that our algorithms achieve low streaming cost due to its excellent anchor view allocation.

I. Introduction

The advent of multiview imaging technologies means that videos from different viewpoints of the same 3D scene can now be captured simultaneously by a system of multiple closely spaced cameras [1]. Furthermore, depth maps, which measure the per-pixel distance between cameras and physical objects, can be captured directly through time-of-flight (ToF) cameras [2], or indirectly through stereo-matching algorithms [3]. When the depth maps are available at the camera viewpoints, virtual views can be synthesized during video playback using texture and depth maps

This work was supported, in part, by Hong Kong Research Grant Council (RGC) General Research Fund (610713), HKUST (FSGRF13EG15) and The Hong Kong Innovation and Technology Fund (UIM/246).

of the closest sandwiched camera views (the so-called *anchor views*) via depth-image-based rendering (DIBR) [4]. The ability of users to synthesize and display any virtual view is called *free viewpoint video*. It enables a 3D visual effect known as *motion parallax*: a viewer's detected head movements can trigger correspondingly shifted video views on his/her 2D display [5]. It is well known that motion parallax is the strongest cue in human's perception of depth in a 3D scene and enhances the immersive visual experience [6].

In *live* free viewpoint video streaming, texture and depth videos from multiple viewpoints in the same 3D scene can be real-time encoded at a server into separate streams before delivery to interested users. The users can choose to look at the recorded camera views or at virtual views arbitrarily positioned between the camera views.

We consider a collaborative peer-to-peer (P2P) sharing system where the users share their anchor views with each other. Users may switch to any virtual viewpoints at will during their streaming session. Each of them obtains two camera views as the left and right anchors by pulling either directly from the live streaming source, or indirectly from the other users. As long as the virtual viewpoint of interest is in *between* a user's two received anchor views, he can use the same set of anchor views to synthesize the virtual viewpoint. However, if the viewpoint moves outside the viewing range bounded by the two current anchor views (i.e., outside the *anchor window*), the user has to obtain new anchor views so as to sandwich the virtual view. Given that the currently subscribed anchor views need to be reselected to accommodate users' newly chosen virtual views, the peer network needs to be reconfigured, which may incur some overhead.

Figure 1 shows an example of our collaborative free viewpoint live streaming system. The live streaming source encodes in real time all captured videos with different camera viewpoints of the 3D scene. Due to bandwidth and/or cost constraints, the source transmits only a subset of these views (Views 5, 10 and 15 in the figure) to a pool of clients for their sharing. Each client may interact with the free viewpoint video by choosing any viewpoint of his/her interest at will over time. For example, at a particular instant client A is interested in virtual viewpoint 6.5 (labeled as $A(6.5)$ in the figure). He obtains camera view 5 from the live source as the left anchor view, and camera view 10 from Client B as the right anchor view, and then synthesizes viewpoint 6.5 given the anchor views 5 and 10. Client D gets anchor views 5 and 10 from A and B, respectively, to synthesize its virtual view of 8.5. It is clear that all clients are getting two anchor views with their virtual viewpoints inside the anchor window. If Client D is to switch to another virtual view, say 12.2, it has to replace its anchor view of 5 by 15, by pulling view 15 from Client C or F. On the other hand, if Client C is to switch to virtual view 7.1, it then has to pull the anchors 5 and 10 from the peer network (from Client A, B, D, E or F) while C is to pull anchor view 15 directly from the live source. For any of the two cases above, the peer network has to be reconfigured.

In this work we study the optimization of the total streaming cost in free viewpoint video streaming. The streaming cost comprises the following three components:

Source access cost: There is an access cost associated with the transmission of an anchor view from the streaming server to the client pool (due to, for examples, server bandwidth, server processing or computation, or network bandwidth). The peers share with each other the anchor views pulled from the source to generate their virtual viewpoints of interest. The sharing cost among the peers is considered to be negligible because the peers are

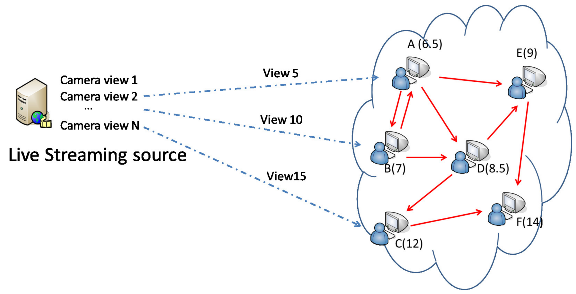


Fig. 1: An example of collaborative live streaming for a free viewpoint video.

within the same local community with high speed network. Due to the source access cost and heterogeneous view popularity, it may not be cost-effective to stream all the anchor views to the client pool.

Video distortion cost: The distortion cost reflects the quality or PSNR of the synthesized virtual views of the video. In general the cost for DIBR synthesized view tends to be larger as the distance from the virtual view to anchor views increases, as experimentally demonstrated and argued using statistical models [7]. We are interested in the total distortion cost for all peers as they select virtual views of different popularities.

Network reconfiguration cost: As the viewpoint of a peer changes over time, he/she may eventually move outside the anchor window. This necessitates the peer to search for some new suppliers for his/her anchor(s) for view synthesis. We assume that there is a network reconfiguration cost due to communication overhead and connection management among peers. The reconfiguration cost reflects the underlying cost and complexity of re-arranging the P2P network, and is a function of the probability for peers to change their anchor views due to view switching interactivity. Such framework is general enough to be applied to any P2P topology. Therefore, the P2P overlay construction, peer group organization and reconfiguration mechanisms are irrelevant to and outside the scope of this paper. There have been extensive studies on these issues. Interested readers may refer to studies in [8], [9].

It is clear from above that the three cost components trade off with each other. On the one hand, it is beneficial for a viewer to request anchor views that tightly “sandwich” its virtual view to reduce the distortion cost. However, this increases both the access cost and reconfiguration cost (due to higher likelihood of a viewpoint falling outside the anchor window). On the other hand, using wider anchor windows may reduce both access and reconfiguration costs. However, this increases the distortion cost.

In this work, we study the anchor view allocation problem to minimize total streaming cost composed jointly of anchor access, video distortion and network reconfiguration. The allocation problem is to optimally select the set of anchor views from the source, so that an appropriate pair of anchor views can be supplied to each peer for synthesis of his chosen virtual view, given the popularity of different virtual views are known. Though much work has been done on peer-to-peer streaming, it is on single-view video which is passive (no interactive view-switching) and hence provides no sensible solution to the anchor view allocation problem that optimizes the tradeoff among different cost components. To the best of our knowledge, this is the first piece of work that addresses the allocation problem for collaborative live streaming of interactive free viewpoint video. Our contributions are as follows:

1) *Problem formulation for interactive free viewpoint live video streaming:* We discuss two representative formulations of the anchor view allocation problem for live free viewpoint video streaming. If the reconfiguration cost is negligible (*e.g.*, peers switch views infrequently or the peer network is a simple topology), we formulate a cost optimization problem named *FLS* which we prove to be solvable in polynomial time. On the other hand, if the network reconfiguration cost is non-negligible, (*e.g.*, in the case of complex P2P network, or large and frequent view-switching by the peers), we formulate the problem as *FLSR*, which is shown to be NP-hard.
2) *Exact optimal algorithm for negligible reconfiguration cost:* We present a polynomial time exact algorithm for *FLS* based on dynamic programming (DP). It works for both of the following cases: i) when the maximum number of anchor views allocated to a peer group cannot be larger than a certain number $B_{\max}$, and ii) when the anchor view access cost is formulated as a cost function (*i.e.*, each anchor view pulled from the source incurs a certain access cost a). Simulation results show that our algorithm makes sensible tradeoffs among the three cost components, and significantly out-performs a traditional sharing approach.
3) *Heuristic algorithms for non-negligible reconfiguration cost:* Since *FLSR* is NP-Hard, we thus present a centralized and locally optimal anchor view allocation heuristic algorithm called *Centralized Grouping* . We further propose a distributed version of our algorithm *Distributed Grouping* with guaranteed convergence, where each peer group independently makes merge-and-split decisions with a well-defined fairness criteria. The simulation results show that our proposed algorithms achieve close-to-optimal cost performance. They substantially outperform a traditional sharing approach.

The outline of the paper is as follows. We first discuss related work in Section II. In Section III, we discuss our system models on the collaborative network, free viewpoints video and view-switching. We present the anchor view allocation problem formulations for negligible and non-negligible reconfiguration costs in Section IV. In Section V, we present an optimal DP algorithm and experimental results for anchor view allocation with negligible reconfiguration cost. We then describe locally optimal solutions with reconfiguration cost in Section VI. Simulation results with reconfiguration cost are presented in Section VII. We conclude in Section VIII.

II. Related Work

We divide the overview of related work into two areas. We first discuss the related work on multiview video streaming and on the high-dimensional media navigation problem. Then we discuss the related work on collaborative streaming of single-view video, as well as game-theoretic analysis of collaborative video streaming.

A. Multiview Video Streaming and High-Dimensional Media Navigation

Much research on multiview video has been focusing on compression (*e.g.*, multiview video coding (MVC) [10], [11]). Streaming strategies and network optimization for multiview video is still a relatively unexplored and new research topic. In their seminal work on multiview video streaming [12], the authors propose a coding and streaming strategy for interactive multiview video to a single user, where only a selected number of captured views are transmitted based on predicted user's view selection. The work in [13] proposes a similar but more advanced coding scheme that is also based on predicted user's selection, but individual code blocks are encoded with a quality proportional to the likelihood that the pixels in the blocks are used in the synthesized image. However, these works have not addressed the problem of collaborative streaming of multiview video, where costs of transmitting video views can be shared among users.

Recent investigations have also studied the problem of loss-resilient multiview videos streaming over error-prone networks. The works in [14] propose to exploit the flexibility provided by reference picture selection (RPS) in H.264 video coding standard for real-time encoded depth video, so that a depth block important to the quality of the synthesized view can be predicted from a transmitted frame further in the past for more reliable decoding [15]. The authors in [16], [17] propose to use distributed source coding (DSC) [18] to enable both periodic view-switching in multiview video and loss resiliency for peers watching the same multiview video synchronized in time but not in view. While loss resiliency in video streaming is an important topic, we consider the orthogonal anchor view selection issue in this paper; we leave the joint loss resiliency and anchor view selection problem for future work.

The study in [19] discusses an *interactive multiview video streaming* (IMVS) video-on-demand scenario, where only a single requested view per client is needed at one time during video playback when the clients may periodically request view-switches. It proposes an efficient coding structure where a captured image can be pre-encoded into multiple versions, so that the appropriate version can be transmitted depending on the currently available content in decoder's buffer, in order to reduce server transmission rate. Later, the work in [20] leverage on the IMVS coding structure in [19] for content replication, so that suitable versions of multiview video segments can be cached in a distributed manner across cooperative network servers. Our current work on anchor view allocation differs from [20] in that: i) we consider the more general *free* viewpoint video, where a client can select and synthesize any intermediate virtual view between two anchor views via DIBR; and ii) we focus on the *live collaborative* streaming scenario, where anchor views can be shared among peers that are synchronized in time but not necessarily in view.

While we focus our study on video view-switching among a feasible set of virtual viewpoints delimited by anchor cameras in a 1D array, there exists a more general *high-dimensional media navigation* problem [21] where the captured media has a much higher number of views than what a user's visual display hardware is capable of displaying at a time (*e.g.*, only one video view at a time can be shown on a conventional 2D display). Thus, interactive browsing by the user is a sensible paradigm, where the user navigates through the large media data by periodically requesting from the server the media subset he/she wants to observe next (e.g., view-switching). Previous works such as [19], [22] provide coding tools to reduce the coding cost of high-dimensional media navigation. We are complimentary to these previous works and we provide algorithms to lower the streaming cost of high-dimensional media among peers interested in the same content. This provides an important building block to the general high-dimensional media navigation problem.

B. Collaborative Video Streaming and Game-Theoretic Analysis

There has been a large body of work on collaborative streaming, addressing different aspects of the problem such as topology construction, scheduling, capacity, security and deployment, etc. The papers in [8], [9], [23]–[26] study the structure and organization of streaming overlays. The works of [27] investigate the maximum theoretical streaming capacity (streaming rate) that can be achieved, given a peer uplink bandwidth constraint or a node degree bound. All these works study *single-view* video streaming, which is *passive* in nature (i.e., no view-switching), and the results cannot be applied to live free viewpoint video streaming, which is *interactive* in nature (i.e., each viewer can freely select a virtual view from which to observe the 3D scene), and where anchor view selection is a critical and challenging issue.

There has been little work studying multiview streaming over P2P network. The work of [28] proposes a scheduling algorithm that allows peers to frequently compute the scheduling of multiview segments. The paper [29] studies the problem of achieving low view-switching delay by organizing viewers of different views together. These works essentially treat the multiview video streaming as multiple single-view video, and it is not clear how to extend them to the live *free* viewpoint streaming problem where the anchor view selection and its effect on virtual view distortion need to be carefully considered.

Collaborative video streaming has also been studied using non-cooperative game-theoretic approaches, where users are often modeled as selfish and rational, and they seek to maximize their own payoff. Thus, the challenge is to design cooperation incentives that encourage users to help each other by sharing their resources. For example, the work in [30] consider single-view video streaming systems where users receive and watch the same stream. The work in [31] consider the more challenging problem of high-dimensional multi-view video streaming and study incentive strategies when users watch different but correlated video streams. All these works use tools from competitive game theory to study the *individual behavior* of each user. In our work, users cooperate with each other to share the access cost in downloading anchor views from the streaming server, and the main issue is on the fair allocation of the cost so that cooperation improves everyone's utility. For this game, cooperative game models are more appropriate. To the best of our knowledge, our work is the first piece of work on collaborative live streaming of interactive free viewpoint video.

It is important to note that we focus on the optimization of the anchor view allocation among peers. After the left and right anchor views for each user are determined, the overlay can be arranged into either a push-based or a pull-based structure for stream distribution. There are various overlay arrangement methods that can be directly applied here (see, for examples, [32]). In addition, issues regarding peer joins and leaves are studied in various works of both single view and multi-view streaming [33]–[35]. Our work is orthogonal to them, and the above approaches can be applied in our system.

III. System Models

In this section, we describe the models that we use in our analysis and in the design of our algorithm for collaborative free viewpoint video streaming. We first present the network model for the streaming server and the peers, followed by the free viewpoint video model that we use in this paper. Finally, we describe how we model user's interactive behavior during view-switching.

A. Network Model

Our free viewpoint video distribution network model consists of only two nodes: S is the live streaming source, and G is a single node representing the group of collaborative peers. (If the peer group is too large, the management cost and control overhead become high. In this case, the large peer group may be sub-divided into smaller ones. Dividing a large peer group into smaller ones is an orthogonal problem and would not be considered here.)

All camera views are generated at the streaming source S and synchronized in time. The connection between the server S and the peer group G may be modeled as a *hard* constraint; *i.e.*, the number of anchor views simultaneously pulled from S by G cannot exceed a pre-defined value $B_{\max}$. This is the case for a local community of local users who are inter-connected by a high-speed network, but are connected to the server via a slower common bottleneck link [36]. Alternatively, the connection between the server S and the peer group G may be modeled as a *soft* constraint, *i.e.*, each anchor view pulled by G induces a cost a in the total cost function. This is the case when the server S charges a fixed price for each additional anchor view the peer group G subscribes to. The linear relationship between the "access cost" and the number of anchor views is based on the assumption that the cost of a typical CDN service is linear with respect to the bandwidth consumption; the video source charges a certain fixed monetary amount per output video stream to users, which is reasonable given most video streaming systems cost models in practice. We will consider these two different connection constraints later in the problem formulation.

B. Free Viewpoint Video Model

Let $\mathcal{V} = \{1, 2, ..., V\}$ be the discrete set of *captured views* for V equally spaced cameras in a 1D array as done in [1] and other works. Each camera captures both a texture map (RGB image) and a depth map (per-pixel physical distances between objects in the 3D scene and camera) at the same resolution. The texture map from an intermediate *virtual view* between any two cameras can be synthesized using texture and depth maps of the two camera views (*anchor views*) via a depth-image-based rendering (DIBR) technique like 3D warping [4]. DIBR essentially maps texture pixels in the anchor views to appropriate pixel locations in a virtual view; such locations are derived from the corresponding depth pixels in the anchor views. Disoccluded pixels in the synthesized view — pixel locations that are occluded in the two anchor views — can be filled in using depth-based inpainting techniques [37], [38]. Because inpainting offers only a best-guess solution, the larger the disoccluded region, the lower the image quality of the synthesized view in general.

More specifically, let u be the virtual view that a peer currently requests for observation. We consider that $u = 1 + k/K$, $k = \{0, \ldots, (V-1)K\}$, for some large pre-determined constant K. In other words, u belongs to an ordered discrete set of intermediate viewpoints — the set of views between (and including) camera views 1 and V, spaced apart by integer multiples of distance $1/K$. Clearly, the set of views approaches a continuum as K increases. (Although we consider equally spaced virtual views for ease of exposition, our analysis and algorithms can be easily generalized to uneven virtual view spacing as well.) A distribution function q_u describes the fraction of peers in the group who currently request the virtual view u. Any virtual view u can be synthesized using left and right anchor views denoted as v_u^l and v_u^r, respectively, where $v_u^l, v_u^r \in \mathcal{V}$ and $v_u^l \leq u \leq v_u^r$. Note that v_u^l and v_u^r do not have to be the closest captured camera views to u.

When texture and depth maps of multiple views of a free viewpoint video are captured at the source, they are compressed and encoded by the encoder at the server before network transmission. In our work on anchor view allocation for collaborative free viewpoint video streaming, we assume that the source coding problem solved by the server is orthogonal to our anchor view allocation problem solved by the peers. In other words, we consider that the compressed camera views (generally of good quality) are given at the server, and focus on minimizing the additional distortion due to the synthesis of virtual views, based on these compressed camera views.

We can model the synthesized view distortion as a sum of the distortion due to lossy source coding of anchor view frames (affecting directly the quality of rendered pixels), and the distortion due to virtual view synthesis (influencing the sizes of disocclusion holes). The second distortion depends on the distance between virtual view and anchor view frames. We essentially seek to minimize it by selecting the appropriate set of anchor views, because source coding has already been decided at the server, which is outside of peers' control. Therefore, we consider the optimization of the source coding distortion in the camera views to be orthogonal to our problem.

The distortion of the synthesized view varies with the choices of anchor views. Let $D_u(v_u^l, v_u^r)$ be the distortion function for peers looking at virtual view u, which is synthesized using v_u^l, v_u^r as anchor views.

We make two assumptions regarding the distortion function $D_u(v_u^l, v_u^r)$. First, we assume that further-away anchor views v_u^l, v_u^r cannot induce smaller distortion in the synthesized view, that is

$$\begin{aligned} D_u(v', v_u^r) &\geq D_u(v, v_u^r), \quad \forall v' < v < u. \\ D_u(v_u^l, v) &\leq D_u(v_u^l, v'), \quad \forall u < v < v'. \end{aligned} \tag{1}$$

We call this *monotonicity in anchor view distance* for synthesized view distortion. This is a reasonable assumption in general, since a further-away reference view typically means larger disoccluded regions in the virtual view image. The disocclusion holes can be filled using inpainting algorithms [39], [40], but in general the larger the holes, higher the penalty in reconstruction quality. This monotonicity assumption is also demonstrated experimentally in [41] using a large number of multiview image sequences.

Second, given that the minimum distance τ between the anchor views v_u^l, v_u^r and the synthesized viewpoint u is defined as $\tau = \min\left\{|v_u^l - u|, |v_u^r - u|\right\}$, we assume that a larger τ will not induce a smaller distortion, *i.e.*,

$$\begin{aligned} D_{u_1}(v^l, v^r) &\leq D_{u_2}(v^l, v^r) \;\; \text{if } \tau_1 < \tau_2 \\ \tau_1 &= \min\left\{|v^l - u_1|, |v^r - u_1|\right\} \\ \tau_2 &= \min\left\{|v^l - u_2|, |v^r - u_2|\right\} \end{aligned} \tag{2}$$

We call this *monotonicity in minimum anchor view distance*. This is also reasonable; it is observed empirically that when both anchor views are encoded at the same quality, the worst synthesized view distortion takes place at the middle view [7]. Further, as the minimum view distance τ approaches zero, the synthesized viewpoint is essentially one of the two anchor views with no distortion due to synthesis. This assumption ensures that the synthesized view distortion for $\tau = 0$ is the minimum possible, which agrees with the earlier statement. In the experimental section, we will construct a specific distortion equation that satisfies these two monotonicity assumptions.

Note that we do not consider distortion due to packet loss in our distortion function D_u. We consider packet loss as an orthogonal problem and focus our distortion model on the synthesized view distortion only. There are plenty of works in the literature that study stream reliability and packet loss recovery, e.g., transmission methods such as *automatic retransmission request* (ARQ) or *forward error correction* (FEC) for UDP. Also in the recent years, HTTP-based live streaming is widely used by many state-of-the-art streaming systems, and protocols such as "HTTP Live Streaming" (HLS) and "Dynamic Adaptive Streaming over HTTP" (DASH) efficiently handle packet loss, congestion control and retransmissions issues by using lower layer transport protocols like TCP with persistent packet retransmission. Hence we consider that packet loss are handled by transmission protocols and that views are delivered reliably to the video clients.

C. View-switching Model

In order to model the view-switching behavior of peers, we consider that a peer with virtual view u can switch in the next time instant to any virtual views w's with probability $P_{u,w}$. The view transition probability matrix is denoted by $\mathbf{P}$. For example, if a peer keeps the current view $u = 1 + k/K$ with probability Ω, and switches to any of the two adjacent views with equal probability $(1 - \Omega)/2$, we have the following transition probabilities:

$$P_{1+k/K,w} = \begin{cases} \Omega, & \text{if } \; w = 1 + k/K; \\ (1-\Omega)/2, & \text{if } \; w = 1 + (k \pm 1)/K; \\ 0, & \text{o.w.} \end{cases} \tag{3}$$

Note that the above view-switching model is a multi-state first-order Markov model: the selection of the next view only depends on the current view u. More complicated view-switching models may also be envisaged; for example, authors in [42] have argued that the selection of the next view is based on the current and the previously selected views. For the sake of simplicity, however, we use the multi-state first-order Markov view-switching model in this paper. Extensions to include more general view-switching models like [42] are conceptually straightforward and hence not discussed here.

IV. Anchor Allocation Problems for Negligible and Non-Negligible Reconfiguration Costs

In this section, we present two formulations of the anchor view allocation problem for negligible and non-negligible reconfiguration costs. Figure 2 shows a road-map of our problems and algorithms to be discussed in this paper.

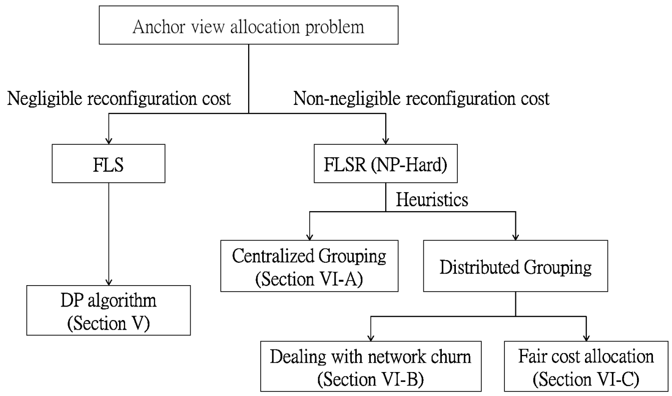


Fig. 2: Road-map for formulations and algorithms.

A. Anchor Allocation with Negligible Reconfiguration Cost

We first consider the case where the reconfiguration cost due to peers' anchor view changes is negligible (due to, for examples, simple peer network protocols or infrequent user view switching). For this case, we formulate the anchor view allocation problem as the *free-viewpoint live streaming* (FLS) problem as discussed below.

Let $\mathcal{V}' \subseteq \mathcal{V}$ be the *purchased set* of captured views selected by the peer group to serve as anchor views to synthesize virtual views. In other words, $\mathcal{V}'$ is streamed from the live source to the user pool. A peer with virtual view u selects left and right anchor views v_u^l and v_u^r from the purchased set $\mathcal{V}'$ to synthesize its target virtual view u. We consider the following *anchor view selection constraint*:

$$v_u^l \leq u \leq v_u^r, \quad v_u^l, v_u^r \in \mathcal{V}' \subseteq \mathcal{V}, \quad \forall u. \tag{4}$$

In words, Equation (4) states that a peer with virtual view u must select from $\mathcal{V}'$ an anchor view v_u^l to the left of u (*i.e.*, $v_u^l \leq u$), and an anchor view v_u^r to the right of u (*i.e.*, $u \leq v_u^r$). The selected anchor views, v_u^l and v_u^r, induce distortion $D_u(v_u^l, v_u^r)$ in the view synthesis process, as discussed in Section III-B. These are our variables to be optimized.

There is an access cost to purchase the set $\mathcal{V}'$ of anchor views by the peer group G. If there is a *hard* connection constraint (or cost budget for the group), we have

$$|\mathcal{V}'| \leq B_{\max}. \tag{5}$$

We label the combinatorial optimization problem with the hard constraint as *FLS-H*, which is to select a subset $\mathcal{V}'$ and anchor views $v_u^l, v_u^r \in \mathcal{V}'$ for each virtual view u, so as to minimize the aggregate distortion of all peers for all virtual views u's, *i.e.*,

$$\min_{\mathcal{V}' \subseteq \mathcal{V}} \; N \sum_u q_u D_u(v_u^l, v_u^r), \tag{6}$$

subject to Constraints (4) and (5), where q_u is the fraction of peers that request view u, where $0 \leq q_u \leq 1$ and $\sum_u q_u = 1$.

One may alternatively consider a *soft* connection constraint, where the total access cost A_{total} for the peer group is proportional to the number of purchased anchor views, *i.e.*, $A_{total} = a\ |\mathcal{V}'|$. This linear model assumes that a certain fixed monetary amount per output video stream is charged by the video source, which is consistent with cost models in many video streaming systems today.

We label the problem with the soft constraint on server connection as *FLS-S*, with the objective of minimizing the sum of the total distortion of all peers for all virtual views u's *and* the total access cost

$$\min_{\mathcal{V}' \subseteq \mathcal{V}} \ N \sum_u q_u D_u(v_u^l, v_u^r) + A_{total}, \tag{7}$$

subject to Constraint (4), where N is the total number of peers in the network. We use the weighted sum method [43], [44] to model these objectives so that the overall streaming cost can be minimized. The exact value of the weight could be modified to fit particular problems.

Note that we are only concerned here with the access cost of camera views in the purchased set $\mathcal{V}'$; the question of how the cost should be fairly distributed to each peer is deferred to Section VI-C.

B. Anchor View Allocation with Reconfiguration Cost

As the video is played back, a peer may switch from a virtual view u to a new view u', where u' may fall outside the range $[v_u^l, v_u^r]$ spanned by the anchor views v_u^l and v_u^r. The network hence needs to be reconfigured to supply the peer with new anchor views. If the reconfiguration cost is non-negligible, the group would tend to choose the anchor views v_u^l and v_u^r that are further apart, so that the likelihood of the virtual view switching outside the range $[v_u^l, v_u^r]$ is low. In this section, we formulate the anchor view allocation problem with reconfiguration cost, termed *free-viewpoint live streaming with reconfiguration* (FLSR, as shown in Figure 2).

The *reconfiguration cost* $S_u(v_u^l, v_u^r)$ depends on the probability that a peer requires new anchor views during the next τ view-switches, given the current virtual view u and the anchor views v_u^l and v_u^r. This probability can be computed as follows. We first define a sub-matrix $\mathbf{P}(v_u^l, v_u^r)$ that contains only the entries $P_{w,z}$'s, defined in Equation (3) with $w, z \in [v_u^l, v_u^r]$. In other words, we keep only entries in $\mathbf{P}$ that correspond to virtual views within the range $[v_u^l, v_n^r]$. Note that the sum of the entries in a row of $\mathbf{P}(v^l, v^r)$ does not need to add up to 1. We can thus write S_u as a simple sum:

$$S_u(v_u^l, v_u^r) = 1 - \sum_w P_{u,w}^{\tau}(v_u^l, v_u^r), \tag{8}$$

where $P_{u,w}^{\tau}(v_u^l, v_u^r)$ is the entry $[u][w]$ in the matrix $\mathbf{P}^{\tau}(v_u^l, v_u^r) = \prod_{t=1}^{\tau} \mathbf{P}(v_u^l, v_u^r)$, which is the $\tau-$step transition probability matrix. Equation (8) states that the reconfiguration cost S_u is one minus the probability that the peer stays within the range $[v_u^l, v_u^r]$ for all τ view-switches.

We first consider the server-peer cost as a hard constraint and formulate the corresponding *FLSR-H* optimization problem. The *FLSR-H* problem is to select a global subset $\mathcal{V}'$ of camera views for the peer group and to select anchor views (v_u^l, v_u^r) for each virtual view u within $\mathcal{V}'$, in order to minimize the sum of the distortion of all peers and the reconfiguration cost weighted by μ, *i.e.*,

$$\min \quad N \sum_u q_u \left(D_u(v^l, v^r) + \mu S_u(v^l, v^r) \right), \tag{9}$$

subject to Constraints (4) and (5). The weight μ is a system parameter that is used to adjust the relative contribution between distortion and the reconfiguration cost. In this formulation section, we on purpose design a general cost function without specifying a particular value for μ, so that our developed algorithms can be applied to a variety of scenarios. In the results section, we will discuss how we choose a sensible μ in different experimental settings.

If the connection costs are considered as a soft constraint, we label the problem as *FLSR-S*. The *FLSR-S* problem is to minimize the sum of the distortion, reconfiguration cost, *and* the total access cost, *i.e.*,

$$\min_{\mathcal{V}' \subseteq \mathcal{V}} \; N \sum_u q_u (D_u(v^l, v^r) + \mu S_u(v^l, v^r)) + A_{total}, \tag{10}$$

subject to Constraint (4).

Both the FLSR-H and FLSR-S problems are NP-hard. The proofs are given in the Appendix.

V. Optimal Solution with Negligible Reconfiguration Cost

Both FLS-H and FLS-S problems can be solved *optimally* in polynomial time via *dynamic programming* (DP). We show here how FLS-S is solved and its performance in Section V-A and V-B, respectively. The algorithm for FLS-H follows similarly in a straightforward manner, and hence is omitted.

A. FLS-S Solution with Dynamic Programming

First, let u_i^l and u_i^r (i stands for initialization) be the leftmost and rightmost virtual views requested by the peer group. Let v_i^l and v_i^r be the corresponding camera views just to the left and to the right of them, *i.e.*,

$$\begin{aligned} v_i^l &= \lfloor u_i^l \rfloor, \qquad u_i^l = \arg\min \; \{u\}, \quad \text{s.t.} \;\; q_u > 0; \\ v_i^r &= \lceil u_i^r \rceil, \qquad u_i^r = \arg\max \; \{u\}, \quad \text{s.t.} \;\; q_u > 0. \end{aligned} \tag{11}$$

v_i^l and v_i^r must be purchased as anchor views in an optimal solution. Define $\varphi(v^l)$ as the minimum cost for all the peers interested in virtual views $u \in [u^l, u_i^r]$, where v^l is an already purchased anchor view, and u^l is the leftmost virtual view in the range $[u_i^l, u_i^r]$ to the right of v^l, *i.e.*,

$$u^l = \arg \min_{u \in [u_i^l, u_i^r] \,|\, u > v^l} u - v^l \tag{12}$$

The optimal solution of FLS-S can be found by a call to $\varphi(v_i^l)$. $\varphi()$ can be recursively calculated as the minimum of

$$\begin{aligned} \varphi(v^l) = \min \Bigg\{ & N \sum_{u^l \le u \le u_i^r} q_u D_u(v^l, v_i^r), \\ & \min_{v^l < v < v_i^r} \Bigg[a \; + N \sum_{u^l \le u \le v} q_u D_u(v^l, v) \\ & \qquad + \; \varphi(v) \Bigg] \Bigg\}, \end{aligned} \tag{13}$$

In words, Equation (13) states that $\varphi()$ is the smaller of:

i) the sum of distortion for synthesized virtual views u's, $u^l \leq u \leq u_i^r$, given that no more anchor views are purchased (and hence v^l and v_i^r are the best anchor views for the synthesis of views $u \in [u^l, u_i^r]$); and

ii) the cost of subscribing to one more anchor view v, $v^l < v < v_i^r$ (*i.e.,* access cost a), plus the sum of distortion for synthesized virtual views u's, $u^l \leq u \leq v$, plus recursive cost $\varphi(v)$ with a reduced virtual view range $(v, u_i^r]$.

Equation (13) follows the divide-and-conquer strategy that is common in dynamic programming (DP), where each recursive call results in a smaller synthesized view range.

The complexity of the solution given by Equation (13) can be analysed as follows. Each time Equation (13) is solved for arguments v^l, the optimal objective value can be stored in entry $[v^l]$ of a DP table Φ, so that any subsequent call to the same sub-problem can simply look up the table. Each minimization in Equation (13) takes $O(V)$ steps to try different anchor view v, and each v requires computation of the sum, which has at most KV terms. Given there are $O(V)$ entries in the DP table, this results in the run-time complexity of $O(V^3K)$.

B. *Experiments and Simulation*

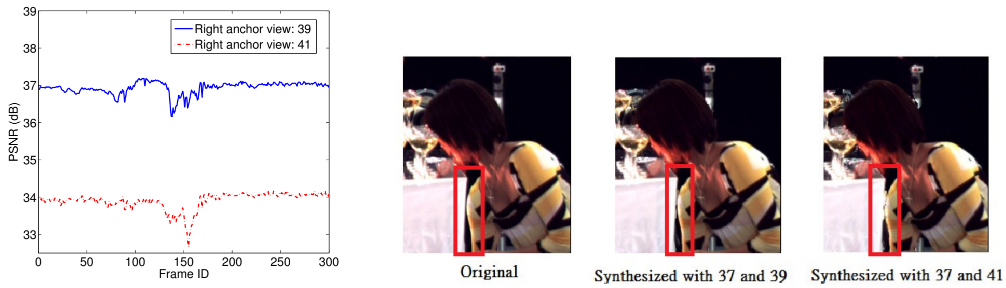
We present here illustrative simulation results for the anchor view allocation problem of FLS-S. As discussed above, the distortion function should monotonically increase with respect to the distance between left and right views, $v^r - v^l$. Furthermore, it should also monotonically increase with respect to the distance between the virtual view and the closer of the two captured views, v^l and v^r. Both of these tendencies are generally observed in empirical multiview data [7], [41]. If the virtual view u is actually one of the anchor views, then the distortion D_u should be zero.

We conduct experiments with view synthesis reference software (VSRS) to study the effect of anchor view selection on the distortion of the synthesized video.

Two multi-view video sequences "Champagne Tower" and "Kendo" (provided by Tanimoto Laboratory, Nagoya University) are used in the experiments. The multi-view video sequences have a resolution of 1280×960 pixels per frame and 30 frames per second. We use the YUV components of the raw camera sequences to synthesize the virtual views. We do not model the source coding error that is due to signal quantization, and only measures the $PSNR$ induced by view synthesis. In "Champagne Tower", we set the left anchor view to camera 37, and use camera 39 or camera 41 as right anchor view to synthesize virtual view 38. As shown in Figure 3a synthesizing the video with anchor views further apart leads to worse PSNR. We also observe in Figure 3b that the virtual view synthesized by closer cameras has less distortion. The results agree with the monotonicity properties of the distortion model.

For simplicity, we consider the following distortion function D_u, which satisfies the two assumptions of monotonicity in Section III-B:

$$D_u(v^l, v^r) = \gamma e^{\alpha_u (v^r - v^l)} \left(e^{\beta_u \times \min(u - v^l, v^r - u)} - 1 \right), \tag{14}$$



(a) PSNR of "Champagne Tower" view 38.

(b) Snapshot of "Champagne Tower" frame 120 view 38.

Fig. 3: PSNR and snapshots of the video at synthesized views.

where γ is the weighting parameter for distortion in the total cost. The rate at which the distortion increases with the distance between anchor views can be adjusted by the parameters α_u and β_u. We calculate the values of α_u and β_u using the MSE distortion of the VSRS-synthesized virtual views in our experiments with the multiview video sequences "Champagne Tower" and "Kendo". The resulting distortion function is used subsequently in the simulations.

We carry out further simulations to study the performance of our algorithms in solving FLS. The simulator is implemented in JAVA, where there are V camera views in the system in total. Between each pair of two adjacent camera views, the same number of virtual viewpoints are generated. The total number of virtual viewpoints is U. There are N peers in the network. Each peer is randomly assigned a virtual viewpoint of interest, and seeks two anchor views to synthesize the target virtual viewpoint in between. The distribution of peers watching different virtual views, i.e., q_u, follows a normal distribution (We have also run simulations using different peer distributions. The results of those simulations are qualitatively the same as what is presented here, and hence are not shown for brevity). The distortion of a peer is calculated according to Equation (14), and the total cost of all peers is calculated according to Equation (7).

We define the price of a camera view as the source access cost of pulling one camera view from the streaming source to the user pool, which is denoted by a. We implement our dynamic programming algorithm, as well as a comparison scheme, *View-independent* approach. In *View-independent* approach, peers independently choose the anchor views that minimize their own distortion without considering peer collaboration on anchor selection. It minimizes the total peer distortion, and the access cost of each anchor view is shared by all users that request it. The baseline parameter values are shown in Table I. We run our simulation on a 64-bit Windows 7 Machine with a Intel i7-2600 CPU. The results are the average of 10 different simulation runs.

Figure 4 shows the total cost (distortion plus access costs) of the peers as a function of the price of camera views. Our dynamic programming (DP) algorithm gives significantly better results than the *View-independent* approach, especially when the price is high. This is because the peers in the DP algorithm can collaboratively select and share the same anchor views to reduce the access cost, achieving a low distortion penalty. As fewer camera views are

TABLE I: Baseline parameters in our simulation.

Parameter	Baseline value
Number of camera views	21
Number of virtual views	200
Number of peers	$2,000$
Price of a camera view: a	5
Distortion weighting parameter: γ	0.01

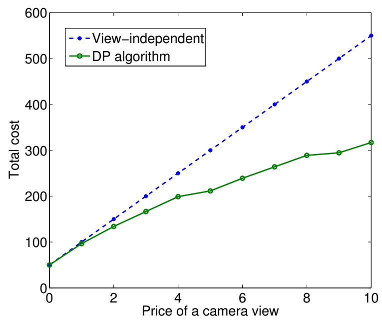


Fig. 4: Total cost versus price of camera views for FLS-S.

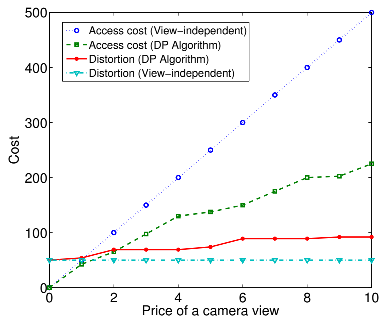


Fig. 5: Cost components versus price of camera views for FLS-S.

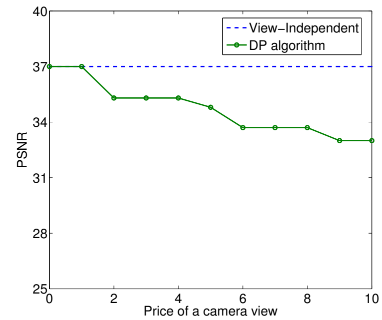


Fig. 6: Average PSNR versus price of camera views for FLS-S.

pulled from the server, the total cost is low.

Figure 5 shows the cost components of peers as a function of the price of camera views. In the *View-independent* approach, the peers greedily pull anchor views to minimize their own distortion. Therefore it leads to high access cost when the anchor view price is high. On the other hand, when the price of an anchor view increases, our proposed DP algorithm will pull less anchor views from the streaming source in order to reduce the access cost, with a small tradeoff in peer distortion. It clearly makes sensible decision on the anchor view allocation.

Figure 6 shows the average PSNR observed by the peers as a function of the price of camera views. The peer PSNR is calculated using the MSE distortion model in Equation (14). The *View-independent* approach has optimal PSNR since the peers always pull anchor views to minimize their own distortion without considering access cost or network reconfiguration. Our proposed DP algorithm achieves close-to-optimal average PSNR, and at the same time optimizes the source access cost. It balances the cost components and makes the optimal anchor allocation decision for each peer.

VI. View Allocation Algorithms with Non-Negligible Reconfiguration Cost

In this section, we present locally optimal and effective algorithms to address the anchor view allocation problem with non-negligible reconfiguration cost. We first present a centralized and locally optimal algorithm (Centralized Grouping) based on Lloyd's algorithm used in non-uniform scalar quantization [45]. Then we present a distributed algorithm with guaranteed convergence, along with a fair access cost allocation mechanism (Distributed Grouping).

A. Centralized Grouping: Locally Optimal Algorithm

We present here a low-complexity centralized optimization algorithm that converges to a locally optimal solution for the NP-hard FLSR problem. We first observe that, for a given subset of camera views $\mathcal{V}' \subseteq \mathcal{V}$ and a given access cost, a peer interested in the virtual view u can *independently* select v_u^l and v_u^r from $\mathcal{V}'$ in order to minimize its own sum of distortion and reconfiguration cost given by $D_u(v^l, v^r) + \mu S_u(v^l, v^r)$. This may potentially lead to a better global solution. In other words, a solution cannot be globally optimal if a peer of virtual view u can find a lower sum of distortion and reconfiguration cost by choosing a different left or right anchor views from the same purchased set $\mathcal{V}'$. We formalize this necessary condition for global optimality with the following lemma.

Lemma 1: If $\mathcal{V}'$, v_u^l's and v_u^r's are a set of optimal variables, then peer(s) with any virtual view u cannot switch from a selected left anchor view $v = v_u^l$ to another anchor view $v' \in \mathcal{V}'$ and lower the overall cost. $\square$

The above Lemma also holds for changing the right anchor view to lower the overall cost.

While the first lemma is concerned with switching of anchor views within a fixed subset $\mathcal{V}'$ of camera views, we can similarly construct a second Lemma about the replacement of a selected camera view $v \in \mathcal{V}'$ by another view $v' \notin \mathcal{V}'$.

Lemma 2: If $\mathcal{V}'$, v_u^l's and v_u^r's is a set of optimal variables, then one cannot replace a selected camera view $v \in \mathcal{V}'$ with an unselected view $v' \notin \mathcal{V}'$, so that peers of views u's that currently use view v as anchor view (i.e. $v_u^l = v$ or $v_u^r = v$), switch to v' as anchor view and lower the overall cost. $\square$

The above two Lemmas are analogous to the two necessary conditions in optimizing non-uniform scalar quantization (SQ) in signal processing [45]. SQ is the problem of quantizing a large number of samples in $\mathcal{R}^1$ space into k Voronoi regions for compact representation, so that only $\lceil \log k \rceil$ bits are required to represent a sample with minimal distortion. The first necessary optimal condition for SQ is that each sample is represented by the Voronoi region whose centroid has the minimum distance to itself (minimum distortion). This is similar to our first Lemma. In the second optimal condition for SQ, each Voronoi region can freely estimate a centroid that minimizes the sum of distances to all samples in the region. This is similar to our second Lemma.

Due to the similarity of our problem to SQ, we can employ a modified version of the famed Lloyd's algorithm to effectively solve our anchor view allocation problem. We call our algorithm the *Centralized Grouping* algorithm.

In particular, for the FLSR-H problem, we first select the leftmost and rightmost camera views from the server, and then a total number of $(B_{\max} - 2)$ camera views are randomly pulled in between. For each peer, we identify the "best" anchor views (chosen from $B_{\max}$ selected camera views) that minimize the sum of distortion and reconfiguration cost. Similar to the Lloyd's algorithm, we then iteratively adjust the positions of $(B_{\max} - 2)$ camera views to reduce the total cost of all peers in the group. In each iteration, we go through each one of the $(B_{\max} - 2)$ camera views, calculate the new total costs if we shift the camera view one step towards its left or right. If the new total cost is lower than the original one, we substitute the camera view with the one to its left (or right) respectively. The algorithm stops when the total cost of peers cannot be further reduced. It is guaranteed to converge since the total cost only decreases at each iteration.

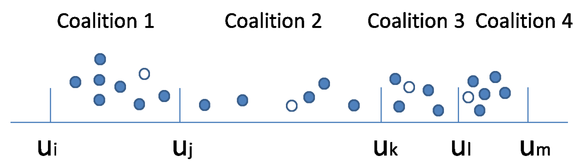


Fig. 7: Coalition of peers.

Finally, for the FLSR-S problem, we run the above procedure $(V-1)$ times with $B_{\max}=2$ to V, and then choose the optimal $\mathcal{V}'$ that yields the minimum total cost that includes distortion, reconfiguration and access costs.

B. *Distributed Grouping: Coalition Maintenance*

The centralized algorithm presented above is able to find a locally optimal FLSR solution by assigning anchor views to each peer. The solution is suitable when there is a central controller, and the network is not large or not highly dynamic (a dynamic system is a system with frequent peer arrivals and departures, and many view switches). Like in any centralized systems, the average runtime for "Centralized Grouping" increases with the number of peers. In this section, we present a simple, adaptive and distributed algorithm for collaborative sharing of anchor views. Our distributed algorithm is adaptive to dynamic networks and scales well to large networks with peer churns (i.e., with peer arrivals and departures). We call this the *Distributed Grouping* algorithm.

In a peer group, the peers watching the same virtual views or adjacent ones are organized into "coalitions". Figure 7 shows an example of how peer coalitions are formed, where $u_i, u_j, \dots, u_m$ are virtual views. Peers watching virtual views between u_i and u_j are organized into a coalition, say, Coalition 1. All peers that belong to the same coalition C use the *Centralized Grouping* algorithm proposed in the previous section to find the optimal set of anchor views that minimizes the total cost (due to distortion, access and reconfiguration as defined in Equation (10)) for all users in the coalition C, and let L_C be the corresponding minimum total cost. All peers in the same coalition share these anchor views and thus access costs. There is a leader peer (marked in white in Figure 7) in each coalition, who keeps track of the number of peers watching each virtual view and of the total cost of the whole coalition. It periodically exchanges information with the two neighboring coalitions on each side. Two neighboring coalitions may merge into a new bigger coalition, and a coalition may also split into two if the overall cost can be reduced.

Inspired by the split-and-merge coalition formation scheme in [46], we use the following algorithms for peer joins, coalition merge and split, peer leaves and view switching.

Peer join: When a new peer i arrives, it first contacts a *Rendezvous Point* (*RP*) that forwards it to the peer group that i belongs to. If there is an existing coalition C that covers the virtual view that peer i requests in the peer group, *RP* connects i with the leader node of the coalition C. The node i joins the coalition C and starts to pull anchor views from other peers in the coalition. The leader peer in C updates the cost and information of the coalition. However, if the virtual view requested by peer i is not in the range of any coalition, a new coalition will be created, and the peer i becomes the leader of the new coalition. It pulls the anchor views from the streaming server in order to minimize its own costs (distortion and reconfiguration costs).

Coalition merge: The coalition structure adapts to peer churns in order to keep the P2P network optimized. The leader of each coalition periodically exchange information. Let L_1, L_2 be the cost for $C1$ and $C2$ respectively, and L'_1, L'_2 be the optimal cost for peers in $C1$ and $C2$ from the result of the *Centralized Grouping* algorithm run on $C_1 \cup C_2$ if they merge and cooperate. If $L'_1 < L_1$ and $L'_2 < L_2$, the two coalitions C_1 and C_2 are merged. Let V_M be the optimal set of anchor views returned by the *Centralized Grouping* algorithm. Each peer i in the merged coalition adapts to new anchor views v_i^{l*} and v_i^{r*} that give the minimum cost ($v_i^{l*}, v_i^{r*} \in V_m$). The leader who requested the merge becomes the new leader of the merged coalition.

Coalition split: For a big coalition C_M, the leader periodically examines whether splitting into two coalitions leads to lower cost. Let u_m be a virtual view separating C_M into two coalitions C_L, C_R. For each different view u_m, the leader runs the *Centralized Grouping* algorithm on both C_L and C_R. If the combination of optimal costs is smaller than L_m, then C_M is split into C_L and C_R, and a new leader is randomly selected for the newly created coalition.

Peer leave: When a peer i is about to leave, all content sharing between i and its neighbors is stopped, and the leader node updates the cost of the coalition. If the leader node leaves, a new leader is randomly chosen.

View switch: A peer i can change its virtual view in the middle of a streaming session. If the new virtual view is still within the range of the coalition, the peer i can still pull anchor views from other peers and synthesize the new view. There is no change in the overlay structure. However, if the new virtual view goes out of the range of the coalition, the peer leaves the current coalition and joins (or creates) a new coalition. It follows the same process as in the situation where peers join or leave the system.

C. Distributed Grouping: Fair Cost Allocation

In the above *Distributed Grouping* algorithm, two neighboring coalitions will merge if this reduces the total cost of all users in both coalitions, and a coalition will split into two if such reduces the total cost of all users in the coalition. Thus, it targets the overall system performance optimization with the assumption that all users are altruistic and willing to sacrifice their own performance to lower the overall system cost.

As peers in P2P networks are selfish and rational, they are willing to cooperate if and only if such cooperation helps to improve their utilities (i.e., if it reduces their cost). Thus, an important issue is to achieve fairness in peer cooperation; we need to study mechanisms to lower each user's cost in addition to minimizing the total cost of the entire P2P network. As such, no user is willing to deviate from the proposed solution, and the constructed overlay P2P network is stable. In our collaborative live free viewpoint video streaming problem, one possibility to address the above is to study the fair allocation of the cost among peers in a coalition; and coalitional game theory provides an ideal tool to provide fair rules for cost reduction via cooperation [47].

Consider a coalition $C = \{1, 2, \cdots, n\}$ with n peers who watch neighboring views and share the anchor views and the access cost. For a subgroup of users $S \subseteq C$ watching nearby views, let L_S be the total cost of peers in

S if they decide to cooperate with users in S only. Given the coalition C with its n members and the minimum total cost L_C determined by the centralized grouping algorithm, an *allocation* vector $\mathbf{x} = [x_1, x_2, \cdots, x_n]$ divides the total cost L_C among its n members, where x_i is the cost assigned to user i and $\sum_{i \in C} x_i = L_C$. Note that, from Section III-B and Equation (8), users' view distortion and reconfiguration costs are determined by the set of pulled views only. Therefore, for a coalition C, given the minimum total cost L_C and the corresponding optimal anchor view set $\mathcal{V}$ determined by the centralized grouping algorithm, the distortion and reconfiguration costs for a user are fixed. Users then discuss how to fairly share the total access cost of $a|\mathcal{V}|$ such that the coalition C and the allocation vector $\mathbf{x}$ are stable.

Given an allocation vector $\mathbf{x}$, we define the *excess* of a subgroup $S \subseteq C$ (with respect to $\mathbf{x}$) as $e(S, \mathbf{x}) = L_S - \sum_{i \in S} x_i$. In this definition, the first term L_S is the total cost of the subgroup S if the peers in this subgroup decide to deviate from the coalition C, to form a new coalition S and to cooperate with peers in S only, but not with others. The second term $\sum_{i \in S} x_i$ is the total cost of S if the peers decide to stay with the coalition C and with the same allocation vector $\mathbf{x}$. Therefore, the excess of the subgroup S with respect to the allocation vector $\mathbf{x}$ is the extra cost incurred to S if the peers deviate from the coalition C and the allocation $\mathbf{x}$ but form a new coalition by themselves. Apparently, if $e(S, \mathbf{x}) > 0$ and forming a new coalition S incurs more cost to peers in S, the subgroup has no incentive to deviate from the coalition C. For a given allocation, if its excesses are all non-negative, then users in C have an incentive to stay in C, and C is a stable coalition. Our goal is to find such stable coalitions and allocation vectors.

Finding such stable allocations is often difficult, and a well-known fair solution is the *nucleolus* [47], [48], which always exists and is unique. It maximizes the excesses in the non-decreasing order, or equivalently, minimizes peers' dissatisfaction in the non-increasing order. That is, it first selects all allocation vectors that maximize the smallest excess (or equivalently, the allocation vectors that minimize the dissatisfaction of the subgroup(s) of peers that gain the least from staying in the coalition C). Then it finds from the selected allocation vectors those that maximize the second smallest excess, and repeats this process until the allocation vector satisfying all the above constraints is unique. Nucleolus is often the desired solution since, if there exist stable allocations, nucleolus is always one of them.

The nucleolus is defined as follows. Given an allocation $\mathbf{x}$, we sort all excesses $\{e(S, \mathbf{x}), \emptyset \neq S \neq C\}$ in the non-decreasing order, and let $\Phi(\mathbf{x})$ be the sorted access vector. The nucleolus η is the unique allocation that lexicographically maximizes Φ over all allocations, that is, $\Phi(\eta) \succ_{lex} \Phi(\mathbf{x}), \forall \mathbf{x} \neq \eta$. Given two vectors $\mathbf{a}$ and $\mathbf{b}$ sorted in the non-decreasing order, $\mathbf{a}$ is said to be lexicographically larger than $\mathbf{b}$ ($\mathbf{a} \succ_{lex} \mathbf{b}$) if in the first component that they differ, that component of $\mathbf{a}$ is larger than the one of $\mathbf{b}$.

To compute the nucleolus, we follow the above definition and solve a sequence of linear programs as follows [48]. We first solve the following problem

$$\begin{aligned} (LP_1) \qquad & \max \ \epsilon \\ \text{s.t.} \qquad & \sum_{i \in C} x_i = L_C, \end{aligned}$$

$$\sum_{i\in S} x_i \le L_S - \epsilon, \ \forall S \neq \emptyset, S \neq C, \tag{15}$$

which maximizes the smallest excess. Let ϵ_1 be the optimal solution of (LP_1), which is the maximal smallest excess, and let S_1 be the collection of all subgroups whose excesses are equal to ϵ_1. We then solve

$$\begin{aligned}
(LP_2) \quad & \max \ \epsilon \\
\text{s.t.} \quad & \sum_{i\in C} x_i = L_C, \\
& \sum_{i\in S} x_i = L_S - \epsilon_1, \ \forall S \in S_1, \\
& \sum_{i\in S} x_i \le L_S - \epsilon, \ \forall S \notin S_1,
\end{aligned} \tag{16}$$

which maximizes the second smallest excess. The second constraint forces the excesses of all subgroups in S_1 to be ϵ_1, the maximum smallest excess found in (LP_1). We continue the same process until there is only one allocation $\mathbf{x}$ that satisfies all the constraints that allocation is the nucleolus.

In *Distributed Grouping*, we apply the above procedure to compute the nucleolus for each coalition found by the algorithm described in Section VI-B. Computing the nucleolus involves solving a sequence of linear programs. There are $O(2^n)$ linear programs in total. Studies have shown that the number of linear programs can be reduced to $O(n)$ without increasing their size [49]. The complexity of computing the nucleolus can be further reduced by considering all peers watching the same virtual view as one single node.

VII. Illustrative Simulation Results with Non-Negligible Reconfiguration Cost

A. Simulation Environment, Comparison Schemes and Metrics

We carried out simulations to evaluate the performance of our proposed *Centralized Grouping* (given by Section VI-A) and *Distributed Grouping* (given by Sections VI-B and VI-C). In our simulation, the probability of each peer staying at the same view is Ω, while the total view switches of each peer is τ. The reconfiguration cost for each peer can be calculated according to Equation (8), and the total cost for all peers are calculated with Equation (10). We set the view switch parameters as follows: $\Omega = 0.6$, $\tau = 0.6$, $\mu = 0.1$. The set of parameters are chosen so that the reconfiguration cost in Equation (14) is in the same scale as the distortion cost. In this way, we explore how our algorithms allocate anchor views given the tradeoff between distortion and reconfiguration cost. Unless otherwise stated, we use the same distortion model and simulation environment as in Section V-B.

We compare our algorithms with the following schemes:

- *Optimal*: The optimal solution is obtained by exhaustive search;
- *View-independent*: In this approach, the peers independently optimize their costs (distortion and reconfiguration). It is similar to the *View-independent* approach in FLS except that peers minimize their own total cost instead of distortion.

We evaluate the performance of our proposed algorithms using the following metrics:

- *Streaming cost:* We are interested in the total streaming cost, which is the weighted sum of distortion, reconfiguration cost and access cost of all peers. Our algorithms (*Centralized Grouping* and *Distributed*

Grouping) minimize this cost. Besides the total cost, we are also interested in the cost components and distributions among peers.

- *Camera views pulled:* In our algorithms, peers collaboratively pull camera views from the streaming server, and the cost for accessing the camera views are shared. To understand how the algorithms work, we study the evolution of the total number of camera views pulled versus system parameters, as well as the distribution of anchor views and their popularity among peers.

B. *Streaming Cost*

Figure 8 shows the total streaming cost of all peers as a function of camera view prices. The total cost increases with the price of a camera view. This is because a higher view price leads to a higher access cost, and peers tend to use the same anchor views with others so they can share the cost of transmitting common anchor views from the streaming server. This, in turn, increases other cost components, *i.e.*, the distortion and reconfiguration costs. From Figure 8, we see that *Centralized Grouping* performs very close to the global optimal solution. The anchor views can successfully be moved to good positions to minimize the total costs of all peers. *Distributed Grouping* is also very efficient in reducing the total cost, especially when the price of a captured view is high. *Distributed Grouping* does not outperform *View-independent* when the view price is low due to the lack of global information.

Figure 9 shows the cost components of *Centralized Grouping* algorithm. With the increase of view price, the access cost becomes the major component of the total cost. Distortion and reconfiguration costs also increase because peers compromise to sub-optimal anchor views (in terms of distortion and reconfiguration) so that their access costs can be shared with a larger crowd. The cost components of *Distributed Grouping* are qualitatively the same as *Centralized Grouping*, and hence are not shown for brevity.

Figure 10 shows the cost distribution of all peers. The majority of the peers in both *Centralized Grouping* and *Distributed Grouping* have a low overall cost. All peers in *Distributed Grouping* have a similar cost because of the fair cost allocation mechanism within a coalition. On the other hand, in *Centralized Grouping*, the access cost is evenly shared among all peers. Therefore. the peers with large distortion and reconfiguration cost have significantly larger overall cost than others.

We show in Figure 11 the cost versus number of peers. The total cost increases with the number of peers. *View-independent* performs the worst. It has a very high total cost even when the number of peers is low. This is due to the lack of collaboration in anchor view selections. *Centralized Grouping* and *Distributed Grouping* achieve close-to-optimal performance. When there are fewer peers in the system, they tend to use the same anchor views to reduce the access cost, with a penalty in other cost components. When the peer population increases, each peer can choose better anchor views, which leads to a lower distortion and reconfiguration cost, since there are more neighbors to share the access cost.

Figure 12 shows the total cost as a function of the number of camera views. When there are more anchor views, the virtual views that the peers watch become further apart from each other, and hence the peers are less likely to share anchor views due to the high distortion and reconfiguration penalty. Therefore the total cost increases with

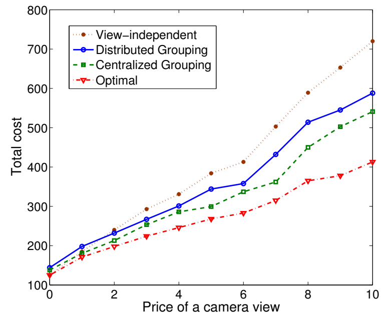


Fig. 8: Total cost versus anchor price.

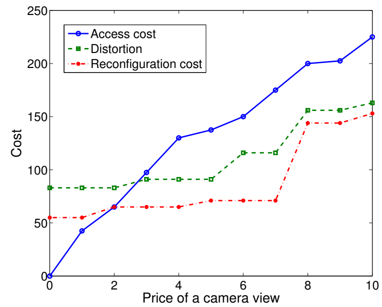


Fig. 9: Cost component versus anchor price.

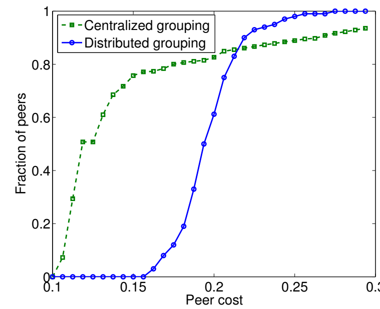


Fig. 10: Cost distribution of peers.

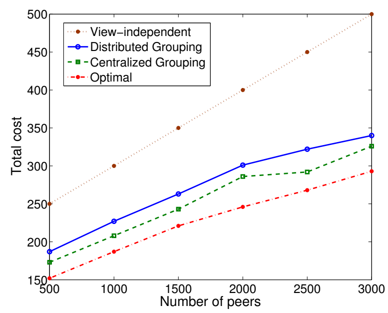


Fig. 11: Total cost versus number of peers.

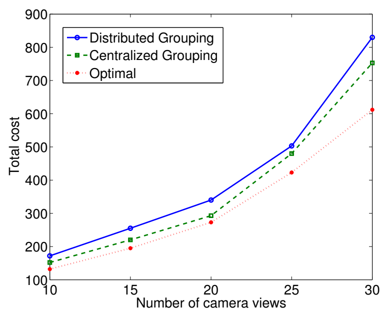


Fig. 12: Total cost versus number of camera views.

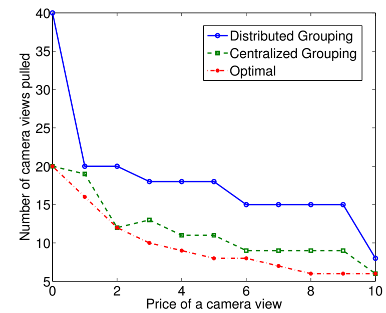


Fig. 13: Number of camera views pulled versus anchor price.

the number of anchor views.

C. Camera Views Pulled

Now we look closer at the distribution of the camera views in the different allocations algorithms. Figure 13 shows the total number of views pulled from the streaming server as a function of the access cost of an anchor view. The number drops with the increase in the price of a camera view. When requesting a captured view from the streaming server becomes expensive, peers tend to seek more cooperation by using the same anchor views and sharing the access cost. Therefore, the total number of camera views pulled from the streaming server becomes smaller. In *Distributed Grouping*, the total number of views pulled could be higher than the total number of camera views since peers only share the access costs within the same coalition, and a captured view could be pulled multiple times by peers from different coalitions.

Figure 14 shows the popularity of camera views in *Centralized Grouping*, i.e., how many times a camera view is used as anchor views by peers. With a small standard deviation of the virtual view distribution, only a few camera views are pulled. The majority of peers watching similar virtual views in the middle use the same anchors (9, 11 and

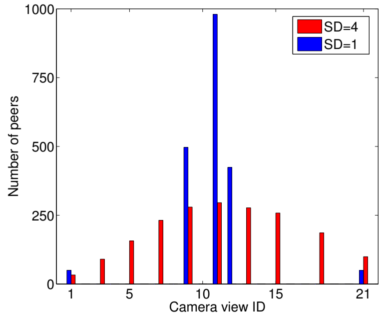


Fig. 14: Anchor view distribution for Centralized Grouping.

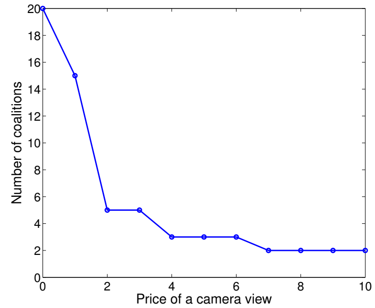


Fig. 15: Number of coalitions formed for Distributed Grouping.

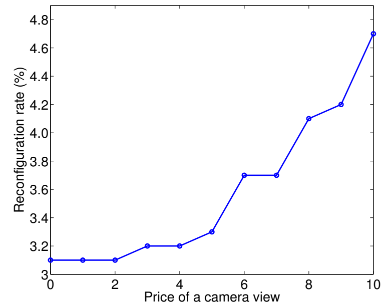


Fig. 16: Reconfiguration rate of peers versus anchor price.

12). Since the rest of the peers are very few in number, they all use the same anchor views (1 or 21) to save access cost with the price of high distortion. When the virtual viewpoints are more "spread out", i.e., standard deviation equals to 4, more camera views are pulled, with more distributed popularity as well. It shows that our algorithms are adaptive to different virtual view distributions of peers. With different distributions, *Centralized Grouping* pulls different numbers of camera views with a different allocation so that the total cost is minimized.

Figure 15 shows the number of coalitions formed by the *Distributed Grouping* algorithm. The number of coalitions drops with the price of a captured view. When the anchor views are expensive, neighboring coalitions are more likely to merge into a bigger one so that the access costs can be shared by more peers. *Distributed Grouping* can efficiently re-arrange the topology to minimize the total cost when the view prices changes.

Figure 16 shows the reconfiguration rate, i.e., the probability that peers need to change their anchor views in *Distributed Grouping*. A peer's virtual view can no longer be synthesized if it goes outside of the range defined by two anchor views in a series of view switches. In this case new anchor views must be obtained, and the P2P network needs to be reconfigured, which leads to system instability. As shown in the Figure, we achieve low probability of anchor change (less than 5%), because we consider the reconfiguration cost directly in our objective function. The reconfiguration rate is higher when the price of a camera view increases, as peers sacrifice their reconfiguration cost to look for better sharing of access cost.

D. Computational time

Figure 17 shows the average run time of the *Centralized Grouping* algorithm as a function of the total number of camera views. It demonstrates the relationship between the computational time of an anchor allocation scheme and the problem complexity. The running time of the algorithm increases with the number of camera views. With more camera views, there are more candidates for the $B_{max} - 2$ pulled camera views, and *Centralized Grouping* needs to go through more iterations before it converges. However, given a reasonable number of camera views, our algorithm achieves very low computational time for a medium-sized P2P system.

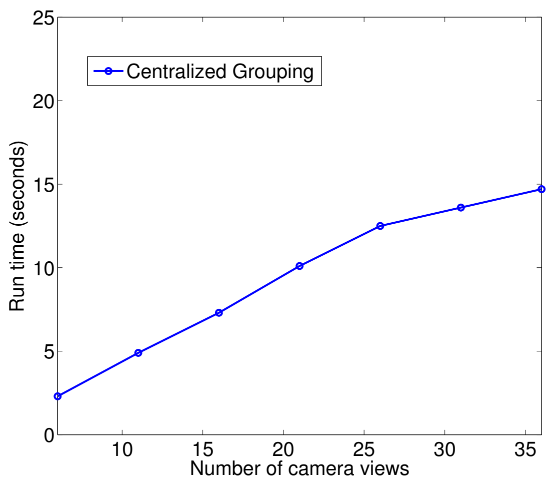


Fig. 17: Average Run time for Centralized Grouping.

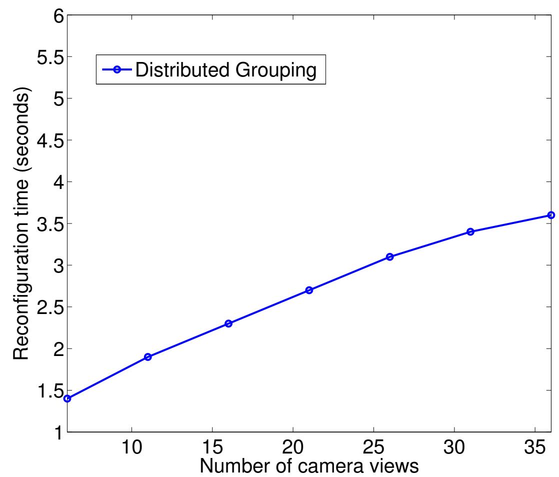


Fig. 18: Average reconfiguration time for Distributed Grouping.

Figure 18 shows the average reconfiguration time of *Distributed Grouping* algorithm as a function of the total number of camera views. The reconfiguration time is defined as the amount of time needed by the coalition leader needs to calculate the new anchor allocation scheme upon peer churns, i.e., peer join, peer leave, view-switch, coalition merge and coalition split. The reconfiguration time mildly increases with the number of camera views. However it is relatively low comparing to the length of the player buffer in real P2P streaming systems, hence this reconfiguration time is not a problem in practice.

VIII. Conclusion

In live free viewpoint streaming, videos from different viewpoints of the same 3D scene are captured by multiple cameras. Peers may select at will different virtual viewpoints, which are synthesized using texture and depth videos of the sandwiched camera views or the so-called anchor views. In this paper we study anchor view allocation problem for collaborative live streaming of free viewpoint video, where peers share with each other their anchor views. There is an access cost to access anchor views from the live source, a distortion cost due to the distance from the virtual views to the anchor view, and a reconfiguration cost due to change of suppliers in the peer network upon view switching. The challenge is how to minimize the total streaming cost by trading off these cost components by allocating anchor views to the peers.

We formulate two problems for anchor view allocation, namely FLS and FLSR, depending on whether the reconfiguration costs are negligible or not. We provide an exact optimal solution based on dynamic programming for FLS. For FLSR, we present a locally-optimal and effective centralized algorithm (Centralized Grouping), and a distributed algorithm with guaranteed convergence (Distributed Grouping). The simulation results show that our proposed algorithms substantially outperform a baseline scheme without collaborative anchor selection. Our results show that collaboration is key in the design of live free viewpoint streaming systems to achieve low distortion and streaming costs.

REFERENCES

[1] T. Fujii, K. Mori, K. Takeda, K. Mase, M. Tanimoto, and Y. Suenaga, "Multipoint measuring system for video and sound—100 camera and microphone system," in *IEEE International Conference on Multimedia and Expo*, Toronto, Canada, July 2006.

[2] S. Gokturk, H. Yalcin, and C. Bamji, "A time-of-flight depth sensor—system description, issues and solutions," in *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, Washington, DC, June 2004.

[3] D. Scharstein and C. Pal, "Learning conditional random fields for stereo," in *IEEE Conference on Computer Vision and Pattern Recognition*, Minneapolis, MN, June 2007.

[4] W. Mark, L. McMillan, and G. Bishop, "Post-rendering 3D warping," in *Symposium on Interactive 3D Graphics*, New York, NY, April 1997.

[5] M. Tanimoto, M. P. Tehrani, T. Fujii, and T. Yendo, "Free-viewpoint TV," in *IEEE Signal Processing Magazine*, vol. 28, no.1, January 2011.

[6] S. Reichelt, R. Hausselr, G. Futterer, and N. Leister, "Depth cues in human visual perception and their realization in 3D displays," in *SPIE Three-Dimensional Imaging, Visualization, and Display 2010*, Orlando, FL, April 2010.

[7] P. Merkle, Y. Morvan, A. Smolic, D. Farin, K. Muller, P. de With, and T. Wiegand, "The effects of multiview depth video compression on multiview rendering," in *Signal Processing: Image Communication*, vol. 24, 2009, pp. 73–88.

[8] N. Magharei, R. Rejaie, and Y. Guo, "Mesh or multiple-tree: A comparative study of live P2P streaming approaches," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, may 2007, pp. 1424 –1432.

[9] X. Lu, Q. Wu, R. Li, and Y. Lin, "On tree construction of super peers for hybrid P2P live media streaming," in *Computer Communications and Networks (ICCCN), 2010 Proceedings of 19th International Conference on*, aug. 2010, pp. 1 –6.

[10] J.-H. Kim, J. Garcia, and A. Ortega, "Dependent bit allocation in multiview video coding," in *IEEE International Conference on Image Processing*, Genoa, Italy, September 2005.

[11] M. Flierl, A. Mavlankar, and B. Girod, "Motion and disparity compensated coding for multiview video," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no.11, November 2007, pp. 1474–1484.

[12] E. Kurutepe, M. R. Civanlar, and A. M. Tekalp, "Client-driven selective streaming of multiview video for interactive 3DTV," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no.11, November 2007, pp. 1558–1565.

[13] D. Florencio and C. Zhang, "Multiview video compression and streaming based on predicted viewer position," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Taipei, Taiwan, April 2009.

[14] B. Macchiavello, C. Dorea, M. Hung, G. Cheung, and W. t. Tan, "Reference frame selection for loss-resilient texture & depth map coding in multiview video conferencing," in *IEEE International Conference on Image Processing*, Orlando, FL, September 2012.

[15] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no.7, July 2003, pp. 560–576.

[16] Z. liu, G. Cheung, and Y. Ji, "Unified distributed source coding frames for interactive multiview video streaming," in *IEEE International Conference on Communications*, Ottawa, Canada, June 2012.

[17] D. Ren, S.-H. G. Chan, G. Cheung, and P. Frossard, "Coding structure and replication optimization for interactive multiview video streaming," *IEEE Transactions on Multimedia*, to appear.

[18] N.-M. Cheung, A. Ortega, and G. Cheung, "Distributed source coding techniques for interactive multiview video streaming," in *27th Picture Coding Symposium*, Chicago, IL, May 2009.

[19] G. Cheung, A. Ortega, and N.-M. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," in *IEEE Transactions on Image Processing*, vol. 20, no.3, March 2011, pp. 744–761.

[20] H. Huang, B. Zhang, G. Chan, G. Cheung, and P. Frossard, "Coding and replication co-design for interactive multiview video streaming," in *mini-conference in IEEE INFOCOM*, Orlando, FL, March 2012.

[21] G. Cheung, A. Ortega, N.-M. Cheung, and B. Girod, "On media data structures for interactive streaming in immersive applications," in *SPIE Visual Communications and Image Processing*, Huang Shan, China, July 2010.

[22] P. Ramanathan, M. Kalman, and B. Girod, “Rate-distortion optimized interactive light field streaming,” in *IEEE Transactions on Multimedia*, vol. 9, no.4, June 2007, pp. 813–825.

[23] D. Ren, Y.-T. H. Li, and S.-H. G. Chan, “On reducing mesh delay for peer-to-peer live streaming,” in *IEEE INFOCOM*. Phoenix, Arizona: IEEE, apr 2008.

[24] ——, “Fast-mesh: a low-delay high-bandwidth mesh for peer-to-peer live streaming,” *IEEE Transactions on Multimedia*, vol. 11, no. 8, Dec. 2009.

[25] X. Jin, K.-L. Cheng, and S.-H. G. Chan, “SIM: Scalable island multicast for peer-to-peer media streaming,” in *Proc. IEEE International Conference on Multimedia and Expo (ICME)*, Toronto, Canada, jul 2006, pp. 913–916.

[26] W.-P. K. Yiu, X. Jin, and S.-H. G. Chan, “Challenges and approaches in large-scale peer-to-peer media streaming,” *IEEE Multimedia Magazine*, vol. 14, no. 2, pp. 50–59, April - June 2007.

[27] S. Liu, M. Chen, S. Sengupta, M. Chiang, J. Li, and P. A. Chou, “P2p streaming capacity under node degree bound,” in *Proceedings of the 2010 IEEE 30th International Conference on Distributed Computing Systems*, ser. ICDCS. Washington, DC, USA: IEEE Computer Society, 2010.

[28] Y. Ding and J. Liu, “Efficient stereo segment scheduling in peer-to-peer 3D/multi-view video streaming,” in *Peer-to-Peer Computing, 2011 IEEE International Conference on*, sept 2011, pp. 182 –191.

[29] Z. Chen, L. Sun, and S. Yang, “Overcoming view switching dynamic in multi-view video streaming over P2P network,” in *3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), 2010*, june 2010, pp. 1 –4.

[30] G. Tan and S. A. Jarvis, “A payment-based incentive and service differentiation mechanism for peer-to-peer streaming broadcast,” *International Workshop on Quality of Service (IWQoS)*, June 2006.

[31] V. H. Zhao and G. Cheung, “Game theoretical analysis of wireless multiview video multicast using cooperative Peer-to-Peer repair,” *IEEE Workshop on Streaming and Media Communications*, July 2011.

[32] M. Zhang, Q. Zhang, L. Sun, and S. Yang, “Understanding the power of pull-based streaming protocol: Can we do better?” *Selected Areas in Communications, IEEE Journal on*, vol. 25, no. 9, Dec. 2007.

[33] H. Liu, X. Liu, W. Song, and W. Wen, “An age-based membership protocol against strong churn in unstructured p2p networks,” in *Proceedings of the 2011 International Conference on Network Computing and Information Security - Volume 02*, ser. NCIS ’11, 2011, pp. 195–200.

[34] X. Meng, X. Chen, and Y. Ding, “Using the complementary nature of node joining and leaving to handle churn problem in p2p networks,” *Comput. Electr. Eng.*, vol. 39, no. 2, pp. 326–337, Feb. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.compeleceng.2012.11.001

[35] Z. Yang, W. Wu, K. Nahrstedt, G. Kurillo, and R. Bajcsy, “Enabling multi-party 3d tele-immersive environments with viewcast,” *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 6, no. 2, pp. 7:1–7:30, Mar. 2010.

[36] P. Sharma, S.-J. Lee, J. Brassil, and K. Shin, “Distributed communication paradigm for wireless community networks,” in *IEEE International Conference on Communications*, Seoul, Korea, May 2005.

[37] K. Oh, S. Yea, and Y.-S. Ho, “Hole-filling method using depth based in-painting for view synthesis in free viewpoint television and 3D video,” in *27th Picture Coding Symposium*, Chicago, IL, May 2009.

[38] I. Daribo and B. Pesquet-Popescu, “Depth-aided image inpainting for novel view synthesis,” in *IEEE International Workshop on Multimedia and Signal Processing*, Saint-Malo, France, October 2010.

[39] I. Ahn and C. Kim, “Depth-based disocclusion filling for virtual view synthesis,” in *IEEE International Conference on Multimedia and Expo*, Melbourne, Australia, July 2012.

[40] S. Reel, G. Cheung, P. Wong, and L. Dooley, “Joint texture-depth pixel inpainting of disocclusion holes in virtual view synthesis,” in *APSIPA ASC*, Kaohsiung, Taiwan, October 2013.

[41] G. Cheung, V. Velisavljevic, and A. Ortega, “On dependent bit allocation for multiview image coding with depth-image-based rendering,” in *IEEE Transactions on Image Processing*, vol. 20, no.11, March 2011, pp. 3179–3194.

[42] X. Xiu, A. Ortega, and J. Liang, “Delay-cognizant interactive multiview video with free viewpoint synthesis,” in *IEEE Transactions on Multimedia*, vol. 14, no.4, March 2012, pp. 1109–1126.

[43] K. Deb, *Multi-objective optimization using evolutionary algorithms*. John Wiley & Sons, 2001, vol. 16.

[44] V. Chankong and Y. Haimes, *Multiobjective decision making: theory and methodology*, ser. North-Holland series in system science and engineering. North Holland, 1983.

[45] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1992.

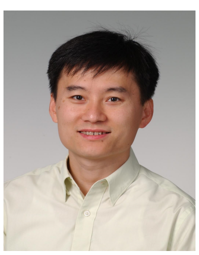
[46] M. D. W. Saad, Z. Han and A. Hjrungnes, "A distributed merge and split algorithm for fair cooperation in wireless networks," *IEEE International Conference on Communications (ICC), Workshop on Cooperative Communications and Networking*, May 2008.

[47] G. Owen, *Game Theory*. Academic Press, 1995.

[48] U. Faigle, W. Kern, and J. Kuipers, "On the computation of the nucleolus of a cooperative game," *International Journal of Game Theory*, vol. 30, no. 1, pp. 79–98, Sept. 2001.

[49] B. Fromen, "Reducing the number of linear programs needed for solving the nucleolus problem of n-person game theory," *European Journal of Operational Research*, no. 98, pp. 626–636, 1997.

Dongni Ren received his Ph.D. candidate at the Department of Computer Science and Engineering in the Hong Kong University of Science and Technology (HKUST), supervised by Prof. Gary Chan. He also received his BEng in Computer Science (Information Engineering) from HKSUT in 2007, and his MPhil in Computer Science from HKUST in 2009. His research interest includes video streaming networks, overlay broadcasting, Video on Demand (VOD), and multi-view/free-viewpoint video technologies.



Dr. S.-H. Gary Chan (S'89—M'98—SM'03) is currently Professor and Undergraduate Programs Coordinator at the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology (HKUST), Hong Kong. He is also the Director of Sino Software Research Institute at HKUST. He received MSE and PhD degrees in Electrical Engineering from Stanford University (Stanford, CA) in 1994 and 1999, respectively, with a minor in business administration. He obtained his B.S.E. degree (highest honor) in Electrical Engineering from Princeton University (Princeton, NJ) in 1993. His research interest includes multimedia networking, wireless networks and mobile computing.

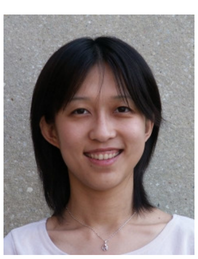
Professor Chan was an Associate Editor of IEEE Transactions on Multimedia (2006-11), and a Vice-Chair of Peer-to-Peer Networking and Communications Technical Sub-Committee of IEEE Comsoc Emerging Technologies Committee. He has been Guest Editors of IEEE Transactions on Multimedia (2011), IEEE Signal Processing Magazine (2011), IEEE Communication Magazine (2007), and Springer Multimedia Tools and Applications (2007). He was the TPC chair of IEEE Consumer Communications and Networking Conference (CCNC) 2010, Multimedia symposium in IEEE Globecom (2007 and 2006), IEEE ICC (2007 and 2005), and Workshop on Advances in Peer-to-Peer Multimedia Streaming in ACM Multimedia Conference (2005).

Professor Chan's research projects on wireless and streaming have received several ICT (Information and Communication Technology) awards in Hong Kong, Pan Pearl River Delta and Asia-Pacific regions due to their commercial impacts to industries (2012, 2013 and 2014). He is the recipient of Google Mobile 2014 Award (2010 and 2011) and Silver Award of Boeing Research and Technology (2009). He has been a visiting professor or researcher in Microsoft Research (2000-11), Princeton University (2009), Stanford University (2008 - 09), and University of California at Davis (1998 - 1999). He was a Co-director of HKUST Risk Management and Business Intelligence program (2011 - 2013) , and Director of Computer Engineering Program at the HKUST (2006 - 2008).

Gene Cheung (M'00—SM'07) received the B.S. degree in electrical engineering from Cornell University in 1995, and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California, Berkeley, in 1998 and 2000, respectively.

He was a senior researcher in Hewlett-Packard Laboratories Japan, Tokyo, from 2000 till 2009. He is now an associate professor in National Institute of Informatics in Tokyo, Japan.

His research interests include image & video representation, immersive visual communication and graph signal processing. He has published over 140 international conference and journal publications. He has served as associate editor for IEEE Transactions on Multimedia (2007–2011) and currently serves as associate editor for DSP Applications Column in IEEE Signal Processing Magazine, APSIPA journal on signal & information processing and SPIE Journal of Electronic Imaging, and as area editor for EURASIP Signal Processing: Image Communication. He serves as the lead guest editor of the special issue on "Interactive Media Processing for Immersive Communication" in IEEE Journal on Special Topics on Signal Processing, target published date March 2015. He currently serves as member of the Multimedia Signal Processing Technical Committee (MMSP-TC) in IEEE Signal Processing Society (2012-2014). He has also served as area chair in IEEE International Conference on Image Processing (ICIP) 2010, 2012-2013, technical program co-chair of International Packet Video Workshop (PV) 2010, track co-chair for Multimedia Signal Processing track in IEEE International Conference on Multimedia and Expo (ICME) 2011, symposium co-chair for CSSMA Symposium in IEEE GLOBECOM 2012, and area chair for ICME 2013. He was invited as plenary speaker for IEEE International Workshop on Multimedia Signal Processing (MMSP) 2013 on the topic "3D visual communication: media representation, transport and rendering". He is a co-author of best student paper award in IEEE Workshop on Streaming and Media Communications 2011 (in conjunction with ICME 2011), best paper finalists in ICME 2011 and ICIP 2011, best paper runner-up award in ICME 2012, and best student paper award in ICIP 2013.



H. Vicky Zhao (M'05) received the B.S. and M.S. degrees from Tsinghua University, Beijing, China, in 1997 and 1999, respectively, and the Ph.D. degree from University of Maryland, College Park, in 2004, all in electrical engineering. She was a Research Associate with the Department of Electrical and Computer Engineering and the Institute for Systems Research, University of Maryland, College Park, from January 2005 to July 2006. Since August 2006, she has been an Assistant Professor with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada. Her research interests include information security and forensics, multimedia social networks, digital communications, and signal processing. Dr. Zhao received the IEEE Signal Processing Society (SPS) 2008 Young Author Best Paper Award. She co-authored the book Multimedia Fingerprinting Forensics for Traitor Tracing (New York: Hindawi, 2005). She is an Associate Editor for the IEEE SIGNAL PROCESSING LETTERS and the Journal of Visual Communication and Image Representation.

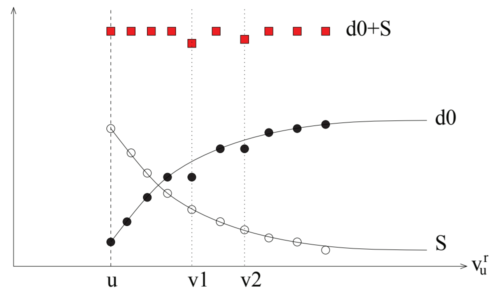


Fig. 19: Cost with different right anchor views, when the left anchor view is fixed.

Pascal Frossard (S'96—M'01—SM'04) received the M.S. and Ph.D. degrees, both in electrical engineering, from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in 1997 and 2000, respectively. Between 2001 and 2003, he was a member of the research staff at the IBM T. J. Watson Research Center, Yorktown Heights, NY, where he worked on media coding and streaming technologies. Since 2003, he has been a faculty at EPFL, where he heads the Signal Processing Laboratory (LTS4). His research interests include image representation and coding, visual information analysis, distributed image processing and communications, and media streaming systems.

Dr. Frossard has been the General Chair of IEEE ICME 2002 and Packet Video 2007. He has been the Technical Program Chair of IEEE ICIP 2014 and EUSIPCO 2008, and a member of the organizing or technical program committees of numerous conferences. He has been an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING (2010-2013), the IEEE TRANSACTIONS ON MULTIMEDIA (2004-2012), and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (2006-2011). He is the Chair of the IEEE Image, Video and Multidimensional Signal Processing Technical Committee (2014-2015), and an elected member of the IEEE Visual Signal Processing and Communications Technical Committee (2006-) and of the IEEE Multimedia Systems and Applications Technical Committee (2005-). He has served as Steering Committee Chair (2012-2014) and Vice-Chair (2004-2006) of the IEEE Multimedia Communications Technical Committee and as a member of the IEEE Multimedia Signal Processing Technical Committee (2004-2007). He received the Swiss NSF Professorship Award in 2003, the IBM Faculty Award in 2005, the IBM Exploratory Stream Analytics Innovation Award in 2008 and the IEEE Transactions on Multimedia Best Paper Award in 2011.

APPENDIX

NP-HARD PROOF OF FLSR

We show that the well-known NP-complete *Minimum Cover* (MC) problem is polynomial-time reducible to a special case of FLSR-H. In MC, a collection $\mathcal{C}$ of subsets of a finite item set $\mathcal{S}$ is given. The decision problem is: does $\mathcal{C}$ contain a *cover* for $\mathcal{S}$ of size at most κ, *i.e.*, a subset $\mathcal{C}' \subseteq \mathcal{C}$ where $|\mathcal{C}'| \leq \kappa$, such that every item in $\mathcal{S}$ belongs to at least one subset of $\mathcal{C}'$?

Consider a special case of FLSR-H where in the optimal solution, all peers use the leftmost camera view 1 as their left anchor view. This is the case if the synthesized distortion for each peer of view u is a local minimum whenever view 1 is used as left anchor, *i.e.*, $D_u(1, v_u^r) \leq D_u(v, v_u^r), \forall v, v_u^r$. Hence all peers will share view 1 as left anchor view, and need to select only the right anchor view to minimize the aggregate cost in Equation (9).

We first map items in set $\mathcal{S}$ to consecutive virtual views u's (each with $q_u = 1/|\mathcal{S}|$) just to the right of leftmost camera view 1. We map subsets in collection $\mathcal{C}$ to camera views v's to the right of the virtual views u's. We next construct the reconfiguration cost $S_u(1, v_u^r)$ by assuming a view-switching probability $\Omega > 0$ in (1) and $\tau = 1$, resulting in a decreasing $S_u(1, v_u^r)$ as a function of v_u^r for all virtual views u's, as shown in Figure 19.

We first set the distortion $D_u(1, v_u^r)$ for peers with virtual views u's such that the aggregate cost is a constant α, *i.e.*, $D_u(1, v_u^r) + S_u(1, v_u^r) = \alpha$. Then, for each item s_i in subset c_j, we reset the distortion $D_u(1, v_u^r)$ (of virtual view u corresponding to item s_i and of anchor view v_u^r corresponding to set c_j) to the distortion $D_u(1, v_u^r - 1)$ with the anchor view $v_u^r - 1$. Note that the distortion function remains monotonically non-decreasing.

Figure 19 shows an example of the aggregate cost for peer with virtual view u, where d_0 is the distortion and S is the reconfiguration cost. Note that $d_0 + S = \alpha$ except for $v_u^r = v_1$ and $v_u^r = v_2$. If an optimal solution to FLSR-H with constraint $V_M = \kappa + 1$ has a total cost less than $|\mathcal{S}|\alpha$, then the selected camera views will correspond to $\mathcal{C}'$ in MC. Hence MC is a special case of FLSR-H. $\Box$

Then We prove that the FLSR-S problem is NP-hard, by reducing the MC problem to a special case of FLSR-S. Following similar construction as in the proof for FLSR-H, we first map items in set $\mathcal{S}$ to virtual views u's (each with $q_u = 1/|\mathcal{S}|$) to the right of leftmost camera view 1, and map subsets in collection $\mathcal{C}$ to camera views v's to the right of the virtual views. Consider again the case where the optimal solution has all peers sharing view 1 as their left anchor view.

We construct the reconfiguration cost $S_u(1, v_u^r)$ as done in the FLSR-H proof. Next, we identify the smallest $S_u(1, v)$ for all u's and v's for which u and v correspond to an item and a subset in the original MC problem, respectively. Let $\delta = S_u(1, v-1) - S_u(1, v)$. We then construct $D_u(1, v)$ to be $1 - S_u(1, v) - \delta$ if the subset corresponding to v contains the item corresponding to u, and $1 - S_u(1, v)$ otherwise. That means that a virtual view covered by a camera view v has a decrease of δ in distortion. Note that by the definition of δ, $D_u(1, v)$ is monotonically non-decreasing. Finally, we define the access cost $a = \delta/(|\mathcal{C}| + 1)$, which means that purchasing all the camera views v's is cheaper than paying for a distortion δ for a virtual view u uncovered by a camera view v.

We now claim that, if the optimal solution to FLSR-S has an access cost smaller than $\kappa\delta/(|\mathcal{C}| + 1)$, then the MC decision problem is positive, and vice versa. This is because under the above construction, FLSR-S can always find a solution that covers all virtual views u's (items in MC) with camera views v's. If the solution requires κ or fewer camera views, then the corresponding subsets will cover all items in $\mathcal{C}$ in MC. $\Box$